

# A Logic for Strategy Updates

Can Başkent

Department of Computer Science, Graduate Center, City University of New York  
cbaskent@gc.cuny.edu    www.canbaskent.net

## 1 Introduction

In game theory, *strategy* for a player is defined as “a set of rules that describe exactly how (...) [a] player should choose, depending on how the [other] players have chosen at earlier moves” [14]. Notice that this definition of strategies is *static*, and presumably constructed *before* the game is actually played.

For example, consider chess. According to Zermelo’s well-known theorem, chess is determined [22]. Then, why would you play chess if you know you will lose (or won’t win) the game? Clearly, if we have logical omniscience (which we don’t), then it is pointless for the player, who is going to lose, to even start playing the game as she knows the outcome already. If we are not logical omniscient, and have only a limited amount of computational power and memory (which we do), then chess is only a perfect information game for God. Therefore, there seems to be a problem. The static, pre-determined notion of strategies falls short analyzing perfect information games. Because, we, people, do not strategize as such even in perfect information games - largely because we are not logically omniscient, and we have limited memory, and computational and deductive power.

While people play games, they observe, learn, recollect and update their strategies *during* the game as well as adopting deontological strategies and goals before the game. Players update and revise their strategies, for instance, when their opponent makes an *unexpected* or *irrational* move. Similarly, sometimes external factors may force the players not to make some certain moves. For instance, assume that you are playing a video game by using a gamepad or a keyboard, and in the middle of the game, one of the buttons on the gamepad

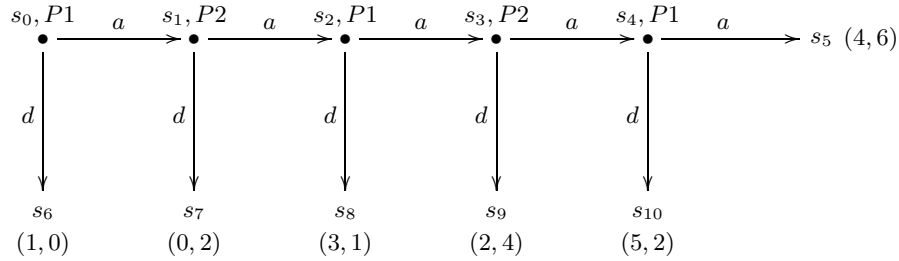


Fig. 1. Centipede game

brakes. Hence, from that moment on, you will not be able to make some moves in the game that are controlled by that button on the gamepad. This is most certainly not part of your strategy. Therefore, you will need to revise your strategy in such a way that some moves will be excluded from your strategy from that moment on. However, for your opponent, that is not the case as she can still make all the moves available to her.

Moreover, in some cases, assumptions about the game or the players may fail as well. For instance, consider the centipede game between two players  $P1$  and  $P2$  (Figure 1). Under the assumption of common rationality, the usual backward induction scheme produces the solution that  $P1$  needs to make a  $d$  move at  $s_0$ . What happens then, if  $P1$  is prohibited or prevented from making  $d$  move at  $s_0$  and onwards right after the beginning of the game (or similarly, if the key on the gamepad that is used to make a  $d$  move is broken)? It means that from a behavioral perspective, the assumption of common rationality is violated and  $P2$  may need to update her strategy *during the game* based on what she has observed. There can be many reasons why  $P1$  may make such a move. The move  $d$  might have been prohibited for  $P1$  right after the game has started, or it may be a taboo for that specific player to make that move, or it may have been simply forbidden or restricted by an external factor (nature, God etc.).

In this paper, we focus on what we call *move updates* where some moves become unavailable during the game. Clearly, there can be considered many other forms of updates, revisions and restrictions that can happen during the game play. For instance, some states may become unavailable in the midst of the game for some players. Moreover, manipulation games where a third party or God/nature affects the outcome of the game are also examples of such games where dynamic strategy analysis is much needed.

Our goal here is to present a formal framework for move based strategy restrictions by extending *strategy logic* (henceforth, SL) which was introduced by Ramanujam and Simon [19]. The rest of the paper is structured as follows. Section 2 provides a short reminder of SL. In Section 3, the framework of SL is extended with strategic move restrictions, and completeness of the resulting logic is shown. Then, we investigate some decision theoretical problems and give a complexity bound for the model checking problem in SL - which was an open problem so far. Finally, we conclude by placing RSL in the context of related work, followed by a conclusion and ideas for future research. The Appendix contains proofs of all propositions and theorems stated in Section 3.

## 2 Strategy logic

In this section, we give a short overview of the strategy logic [12,19]. The focus is on games played between two players given by the set  $N = \{1, 2\}$ , and a single set of moves  $\Sigma$  for both. Let  $\mathbf{T} = (S, \Rightarrow, s_0)$  be a tree rooted at  $s_0 \in S$ , on the set of vertices  $S$ . A partial function  $\Rightarrow: S \times \Sigma \rightarrow S$  specifies the labeled edges of such a tree where labels represent the moves at the states. The extensive form game tree, then, is a pair  $T = (\mathbf{T}, \lambda)$  where  $\mathbf{T}$  is a tree as defined before, and

$\lambda : S \rightarrow N$  specifies whose turn it is at each state. A strategy  $\mu^i$  for a player  $i \in N$  is a function  $\mu^i : S^i \rightarrow \Sigma$  where  $S^i = \{s \in S : \lambda(s) = i\}$ . For player  $i$  and strategy  $\mu^i$ , the strategy tree  $T_\mu = (S_\mu, \Rightarrow_\mu, s_0, \lambda_\mu)$  is the least subtree of  $T$  satisfying the following two natural conditions:

1.  $s_0 \in S_\mu$ ;
2. For any  $s \in S_\mu$ , if  $\lambda(s) = i$ , then there exists a unique  $s' \in S_\mu$  and action  $a$  such that  $s \xRightarrow{a}_\mu s'$ . Otherwise, if  $\lambda(s) \neq i$ , then for all  $s'$  with  $s \xRightarrow{a}_\mu s'$  for some  $a$ , we have  $s \xRightarrow{a}_\mu s'$ .

In other words, in the strategy tree, the root is included, and for the states that belong to the strategizing player, a unique move is assigned to the player, and for the other player, all possible moves are considered. Notice that, in SL, strategies return unique moves. Nevertheless, we would still have a tree even if the strategies are set-valued.

The most basic constructions in SL are strategy specifications. First, for a given countable set  $X$ , a set of basic formulas  $BF(X)$  is defined as follows, for  $a \in \Sigma$ :

$$BF(X) := x \in X \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle a \rangle \varphi$$

Let  $P^i$  be a countable set of atomic observables for player  $i$ , with  $P = P^1 \cup P^2$ . The syntax of strategy specifications is given as follows for  $\varphi \in BF(P^i)$ :

$$Strat^i(P^i) := [\varphi \rightarrow a]^i \mid \sigma_1 + \sigma_2 \mid \sigma_1 \cdot \sigma_2$$

The specification  $[\varphi \rightarrow a]^i$  at player  $i$ 's position stands for “play  $a$  whenever  $\varphi$  holds”. The specification  $\sigma_1 + \sigma_2$  means that the strategy of the player conforms to the specification  $\sigma_1$  or  $\sigma_2$  and  $\sigma_1 \cdot \sigma_2$  means that the strategy of the player conforms to the specifications  $\sigma_1$  and  $\sigma_2$ .

Let  $M = (T, V)$  where  $T = (S, \Rightarrow, s_0, \lambda)$  is an extensive form game tree as defined before, and  $V : S \rightarrow 2^P$  is a valuation function for the set of propositional variables  $P$ . The truth of a formula  $\varphi \in BF(P)$  is given as usual for the propositional, Boolean and modal formulas.

The notion “strategy  $\mu$  conforms to specification  $\sigma$  for player  $i$  at state  $s$ ” (notation  $\mu, s \models_i \sigma$ ) is defined as follows, where  $\mathbf{out}_\mu(s)$  denotes the unique outgoing edge at  $s$  with respect to  $\mu$ .

$$\begin{aligned} \mu, s \models_i [\varphi \rightarrow a]^i & \text{ iff } M, s \models \varphi \text{ implies } \mathbf{out}_\mu(s) = a \\ \mu, s \models_i \sigma_1 + \sigma_2 & \text{ iff } \mu, s \models_i \sigma_1 \text{ or } \mu, s \models_i \sigma_2 \\ \mu, s \models_i \sigma_1 \cdot \sigma_2 & \text{ iff } \mu, s \models_i \sigma_1 \text{ and } \mu, s \models_i \sigma_2 \end{aligned}$$

Now, based on the strategy specifications, the syntax of the strategy logic SL is given as follows:

$$p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle a \rangle \varphi \mid (\sigma)_i : a \mid \sigma \rightsquigarrow_i \psi$$

for  $p \in P$ ,  $a \in \Sigma$ ,  $\sigma \in Strat^i(P^i)$ , and  $\psi \in BF(P^i)$ . We read  $(\sigma)_i : a$  as “at the current state the strategy specification  $\sigma$  for player  $i$  suggests that the move  $a$ ”

can be played". Subsequently, we read  $\sigma \rightsquigarrow_i \psi$  as "following strategy  $\sigma$  player  $i$  can ensure  $\psi$ ". The other connectives and modalities are defined as usual.

We now define the set of available moves at a state  $s$  as  $moves(s) := \{a \in \Sigma : \exists s' \in S \text{ with } s \xrightarrow{a} s'\}$ . Then, based on  $moves$ , we inductively construct the set of enabled moves at state  $s$  in strategy  $\sigma$  as follows.

$$\begin{aligned} - [\psi \rightarrow a]^i(s) &= \begin{cases} \{a\} & : \lambda(s) = i; M, s \models \psi, a \in moves(s) \\ \emptyset & : \lambda(s) = i; M, s \models \psi, a \notin moves(s) \\ \Sigma & : \text{otherwise} \end{cases} \\ - (\sigma_1 + \sigma_2)(s) &= \sigma_1(s) \cup \sigma_2(s) \\ - (\sigma_1 \cdot \sigma_2)(s) &= \sigma_1(s) \cap \sigma_2(s) \end{aligned}$$

The truth definition for the strategy formulas are as follows:

$$\begin{aligned} M, s \models \langle a \rangle \varphi &\quad \text{iff } \exists s' \text{ such that } s \xrightarrow{a} s' \text{ and } M, s' \models \varphi \\ M, s \models (\sigma)_i : a &\quad \text{iff } a \in \sigma(s) \\ M, s \models \sigma \rightsquigarrow_i \psi &\quad \text{iff } \forall s' \text{ such that } s \Rightarrow_\sigma^* s' \text{ in } T_s|_\sigma, \\ &\quad \text{we have } M, s' \models \psi \wedge (\mathbf{turn}_i \rightarrow \mathbf{enabled}_\sigma) \end{aligned}$$

where  $\sigma(s)$  is as before, and  $\Rightarrow_\sigma^*$  denotes the reflexive transitive closure of  $\Rightarrow_\sigma$ . Furthermore,  $T_s$  is the tree that consists of the unique path from the root ( $s_0$ ) to  $s$  and the subtree rooted at  $s$ , and  $T_s|_\sigma$  is the least subtree of  $T_s$  that contains a unique path from  $s_0$  to  $s$  and from  $s$  onwards, for each player  $i$  node, all the moves enabled by  $\sigma$ , and for each node of the opponent player, all possible moves. The proposition  $\mathbf{turn}_i$  denotes that it is  $i$ 's turn to play. Finally, define  $\mathbf{enabled}_\sigma = \bigvee_{a \in \Sigma} (\langle a \rangle \top \wedge (\sigma)_i : a)$ . Now, we give the axioms of SL.

- All the substitutional instances of the tautologies of propositional calculus
- $[a](\varphi \rightarrow \psi) \rightarrow ([a]\varphi \rightarrow [a]\psi)$
- $\langle a \rangle \varphi \rightarrow [a]\varphi$
- $\langle a \rangle \top \rightarrow ([\psi \rightarrow a]^i : a)_i$  for all  $a \in \Sigma$
- $[\mathbf{turn}_i \wedge \psi \wedge ([\psi \rightarrow a]^i : a)] \rightarrow \langle a \rangle \top$
- $\mathbf{turn}_i \wedge ([\psi \rightarrow a]^i : c) \leftrightarrow \neg \psi$  for all  $a \neq c$
- $(\sigma + \sigma')_i : a \leftrightarrow (\sigma : a)_i \vee (\sigma' : a)_i$
- $(\sigma \cdot \sigma')_i : a \leftrightarrow (\sigma : a)_i \wedge (\sigma' : a)_i$
- $\sigma \rightsquigarrow_i \psi \rightarrow [\psi \wedge \mathbf{inv}_i^\sigma(a, \psi) \wedge \mathbf{inv}_{-i}^\sigma(\psi) \wedge \mathbf{enabled}_\sigma]$

Here,  $\mathbf{inv}_i^\sigma(a, \psi) = (\mathbf{turn}_i \wedge (\sigma)_i : a) \rightarrow [a](\sigma \rightsquigarrow_i \psi)$  which expresses the fact that after an  $a$  move by  $i$  which conforms to  $\sigma$ , the statement  $\sigma \rightsquigarrow_i \psi$  continues to hold, and  $\mathbf{inv}_{-i}^\sigma(\psi) = \mathbf{turn}_{-i} \rightarrow \odot(\sigma \rightsquigarrow_i \psi)$  states that after any move of  $-i$ ,  $\sigma \rightsquigarrow_i \psi$  continues to hold. Here,  $\odot \varphi \equiv \bigvee_{a \in \Sigma} \langle a \rangle \varphi$  and  $\odot \varphi \equiv \neg \bigcirc \neg \varphi$ .

Now, we discuss the inference rules that SL employs: modus ponens and generalization for  $[a]$  for each  $a \in \Sigma$ . The induction rule is a bit more complex: From the formulas  $\varphi \wedge (\mathbf{turn}_i \wedge (\sigma)_i : a) \rightarrow [a]\varphi$ ,  $\varphi \wedge \mathbf{turn}_{-i} \rightarrow \odot \varphi$ , and  $\varphi \rightarrow \psi \wedge \mathbf{enabled}_\sigma$  derive  $\varphi \rightarrow \sigma \rightsquigarrow_i \psi$ . The axiom system of SL is sound and complete with respect to the given semantics [19].

### 3 Restricted strategy logic

#### 3.1 Basics

Let us now extend SL to *restricted strategy logic*, henceforth RSL, by allowing move restrictions during the game. Recall that our motivation can be illustrated with the example of a gamepad/keyboard which gets broken during the game play disallowing the player to make some certain moves from that moment on.

We denote the move restriction by  $[\sigma!a]^i$  for a strategy specification  $\sigma$ , and move  $a$  for player  $i$ . Informally, after the move restriction of  $\sigma$  by  $a$ , player  $i$  will not be able to make an  $a$  move. We incorporate restrictions in RSL at the level of strategy specifications. In SL, recall that strategies are functions. Therefore, they only produce one move per state. However, our dynamic take in strategies cover more general cases where strategies can offer a *set* of moves to the player. Thus, in RSL, we define strategy  $\mu^i$  as  $\mu^i : S^i \rightarrow 2^\Sigma$ . By  $\mathbf{outr}_{\mu^i}(s)$  we will denote the set of moves returned by  $\mu^i$  at  $s$ . Then, the extended syntax of strategy specifications for player  $i$  is given as follows.

$$\mathit{Strat}^i(P^i) := [\psi \rightarrow a]^i \mid \sigma + \sigma \mid \sigma \cdot \sigma \mid [\sigma!a]^i$$

Notice that the restrictions affect only the player who gets a move restriction. In other words, if  $a$  is prohibited to player  $i$ , it does not mean that some other player  $j$  cannot make an  $a$  move. In other words, if my gamepad/keyboard is broken, it doesn't mean that yours is broken as well.

Once a move is restricted at a state, we will prune the strategy tree removing the prohibited move from that state on. Therefore, given  $\mu^i : S^i \rightarrow 2^\Sigma$ , we define the updated strategy relation  $\mu^i!a : S^i \rightarrow 2^{\Sigma-\{a\}}$ . We are now ready to define confirmation of restricted specifications to strategies. Note that we skip the cases for  $\cdot$  and  $+$  as they are exactly the same.

$$\begin{aligned} \mu, s \models_i [\varphi \rightarrow a]^i & \text{ iff } M, s \models \varphi \text{ implies } a \in \mathbf{outr}_\mu(s) \\ \mu^i, s \models_i [\sigma!a]^i & \text{ iff } a \notin \mathbf{outr}_{\mu^i}(s) \text{ and } \mu!a, s \models_i \sigma \end{aligned}$$

In the sequel, we omit the superscript that indicates the agents, thus, we write  $S$  for  $S^i$ , and  $\sigma!a$  for  $[\sigma!a]^i$  when it is obvious. Given a strategy  $\mu$  and its strategy tree  $T_\mu = (S_\mu, \Rightarrow_\mu, s_0, \lambda_\mu)$ , we define the restricted strategy structure  $T_{\mu!a}$  with respect to an action  $a$ . Once we removed the restricted moves, the updated structure may not be a tree (it may be a forest). For this reason, we take  $(\mu!a, s)$  as the connected component of  $T_{\mu!a}$  that includes  $s$ . Therefore, for a fixed strategy, restrictions may yield different restricted strategy trees at different states. This is perfectly fine for our intuition, because the state of the game where the restriction is made is important. In other words, it matters where my gamepad is broken during the game play. Now, for player  $i$  and strategy  $\mu^i$  and move  $a$ , the restricted strategy tree  $T_{\mu^i!a} = (S_{\mu^i!a}, \Rightarrow_{\mu^i!a}, s', \lambda_{\mu^i!a})$  is the least subtree of  $T$  satisfying the following two conditions:

1.  $s' \in S_{\mu^i!a}$ ;

2. For any  $s \in S_{\mu!a}$ , if  $\lambda(s) = i$ , then for each action  $b \neq a \in \mu!a(s)$ , there exists a unique  $t \in S_{\mu!a}$  such that  $s \Rightarrow_{\mu!a}^b t$ . Otherwise, if  $\lambda(s) \neq i$ , then for all  $t$  with  $s \Rightarrow_{\mu!a}^b t$  for  $b \neq a$ , we have  $s \Rightarrow_{\mu!a}^b t$ .

Note that we introduce such conditions so that an RSL model can easily be considered as a submodel of a SL model with some additional assumptions. We can now make some further observations. By the abuse of the notation, we will use  $\leftrightarrow$  to denote the equivalence of strategy specifications with respect to the conformation relation.

**Proposition 1.** *For any strategy  $\mu$ , state  $s$ , specification  $\sigma$ , and formula  $\psi$ ,  $\mu, s \not\models [[\psi \rightarrow a]!a]$ .*

**Proposition 2.** *For strategy specifications  $\sigma$  and  $\sigma'$ , and move  $a$ ,  $(\sigma \cdot \sigma')!a \leftrightarrow (\sigma!a) \cdot (\sigma'!a)$  and  $(\sigma + \sigma')!a \leftrightarrow (\sigma!a) + (\sigma'!a)$ .*

Restrictions stabilize immediately. For  $n \geq 1$ , we use notation  $\sigma!^n a$  to denote  $n \geq 1$ -consecutive restrictions of  $\sigma$  by move  $a$ . Similarly, we put  $\mu!^n a$  for the corresponding strategy tree  $\mu$ .

**Proposition 3.** *For arbitrary strategy specification  $\sigma$ , move  $a$  and state  $s$ ,  $(\sigma!a)!a \leftrightarrow \sigma!a$ . Moreover, we have  $\sigma!^n a \leftrightarrow \sigma!a$ .*

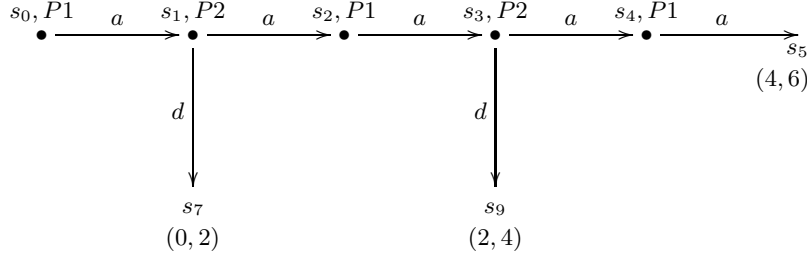
Since restrictions are local and operate by elimination, the order of the restrictions does not matter.

**Proposition 4.** *For any moves  $a$  and  $b$ ,  $(\sigma!a)!b \leftrightarrow (\sigma!b)!a$ .*

### 3.2 A Case Study: The Centipede Game

Let us consider the centipede game (see Figure 1), and see how RSL can formalize it when a restricted strategy specification can change the game after an unexpected (or even irrational) move. Let us call the players  $P1$  and  $P2$ . The set of actions in the centipede game is  $\Sigma = \{d, a\}$  where  $d, a$  mean that the player moves down or across, respectively. Utilities for individual players are indicated by a tuple  $(x, y)$  where  $x$  is the utility for  $P1$ , and  $y$  is the utility for  $P2$ . For the sake of generality, we will not impose any further conditions on the strategies.

In a recent work, Artemov approached the centipede game from a rationality and epistemology based point of view [1]. Now, similar to his approach, we will use symbols  $\mathbf{r}_1$  and  $\mathbf{r}_2$  to denote the propositions “ $P1$  is rational” and “ $P2$  is rational”, respectively. Let us now construct rational strategies  $\mu$  and  $\nu$  for  $P1$  and  $P2$  respectively following the backward induction scheme. At  $s_4$ ,  $P1$  makes a  $d$  move, if she is rational. Therefore, we have  $\mu, s_4 \models [\mathbf{r}_1 \rightarrow d]^{P1}$ . However, at  $s_3$ ,  $P2$  would be aware of  $P1$ ’s possible move at  $s_4$  and the fact that  $P1$  is rational as well, thus makes a  $d$  move if she herself is rational. So, we have  $\nu, s_3 \models [(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)) \rightarrow d]^{P2}$ . Following the same strategy, we obtain the following.



**Fig. 2.** Restricted centipede game-tree

$$\begin{aligned}
\mu, s_2 &\models [\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)) \vee \langle d \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1))) \rightarrow d]^{P1} \\
\nu, s_1 &\models [\mathbf{r}_2 \wedge (\langle a \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)) \vee \langle d \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)))) \vee \langle d \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)) \vee \langle d \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)))))) \rightarrow d]^{P2} \\
\mu, s_0 &\models [(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)) \vee \langle d \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)))))) \vee \langle d \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)) \vee \langle d \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)))))) \vee \langle d \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)) \vee \langle d \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)))))) \vee \langle d \rangle(\mathbf{r}_1 \wedge (\langle a \rangle(\mathbf{r}_2 \wedge (\langle a \rangle \mathbf{r}_1 \vee \langle d \rangle \mathbf{r}_1)))))) \rightarrow d]^{P1}
\end{aligned}$$

Let  $\Diamond\varphi := \langle a \rangle\varphi \vee \langle d \rangle\varphi$ . Then, we have the following statements.

$$\begin{aligned}
\mu, s_4 &\models [\mathbf{r}_1 \rightarrow d]^{P1} \\
\nu, s_3 &\models [\mathbf{r}_2 \wedge \Diamond\mathbf{r}_1 \rightarrow d]^{P2} \\
\mu, s_2 &\models [\mathbf{r}_1 \wedge \Diamond(\mathbf{r}_2 \wedge \Diamond\mathbf{r}_1) \rightarrow d]^{P1} \\
\nu, s_1 &\models [\mathbf{r}_2 \wedge \Diamond(\mathbf{r}_1 \wedge \Diamond(\mathbf{r}_2 \wedge \Diamond\mathbf{r}_1)) \rightarrow d]^{P2} \\
\mu, s_0 &\models [\mathbf{r}_1 \wedge \Diamond(\mathbf{r}_2 \wedge \Diamond(\mathbf{r}_1 \wedge \Diamond(\mathbf{r}_2 \wedge \Diamond\mathbf{r}_1))) \rightarrow d]^{P1}
\end{aligned}$$

Therefore, backward inductively, under the assumption of common rationality, we observe that  $P1$  should make a  $d$  move at  $s_0$ . Furthermore, this can be generalized to many other games.

**Theorem 1.** *Assuming common rationality in RSL framework, backward induction scheme produces a unique solution in games with ordinal pay-offs.*

Argument for the proof of the theorem is quite straight-forward. Even if the strategies may be set valued, and hence return a set of moves per state, the assumption of common rationality forces the player to choose the move which returns the highest pay-off. Since this fact is known among players, by induction, we can show that the solution is unique.

Let us follow the backward induction scheme again with the updated game tree given above. Notice that the specification  $[\mathbf{r}_1 \rightarrow d]^{P1}$  does not conform with  $\mu!d$  (as  $\mu!d, s \not\models [[\mathbf{r}_1 \rightarrow d]^{P1}!d]^{P1}$  for all  $s$ )<sup>1</sup>. Therefore, after the move restriction rational move for  $P1$  is not admissible for her from that point on. Thus,  $\mu!d$  conforms to those specifications that implies an  $a$  move. Thus,  $\mu!d, s_4 \models [\top \rightarrow a]^{P1} \cdot \neg[\mathbf{r}_1 \rightarrow d]^{P1}$ . Therefore, at  $s_3$ , being rational,  $P2$  choses the move with the highest pay-off, and makes an  $a$  move. Thus,  $\nu, s_3 \models \mathbf{r}_2 \wedge (\Diamond\neg[\mathbf{r}_1 \rightarrow d]) \rightarrow a]^{P2}$ .

<sup>1</sup> We slightly abuse the formal language here. For a specification  $\sigma$ , we put  $\mu, s \models \neg\sigma$  if it is not the case that  $\mu, s \models \sigma$ .

Similarly, at  $s_1$ , we have  $\nu, s_1 \models \mathbf{r}_2 \wedge (\Diamond \neg[\mathbf{r}_1 \rightarrow d] \wedge \Diamond(\mathbf{r}_2 \wedge (\Diamond \neg[\mathbf{r}_1 \rightarrow d]))) \rightarrow a]^{P2}$ . Finally, at the root, we have  $\mu!d, s_0 \models [\top \rightarrow a]^{P1}$ . Thus, in the restricted centipede game, clearly,  $P1$  has to make an  $a$  move.

Even if it is not in the scope of this paper, this example also shows that we can view rationality as a *set of restrictions*. Namely, some restrictions forces the players to play irrationally, while some restrictions allow them to play rationally. RSL, in this respect, gives a framework where both rational and irrational strategies can be analyzed.

### 3.3 Axiomatization, Completeness, and Complexity

Before discussing the axiomatization of RSL, we note that the set of available moves *moves* is defined as previously. Then, we define the set of enabled moves for the move restriction operator as  $([\sigma!a]^i)(s) = \sigma(s) - \{a\}$ . Namely, if the move  $a$  is not allowed any more, it should not be enabled.

We now give the syntax of RSL, which is the same as that of SL.

$$p \mid (\sigma)_i : a \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle a \rangle \varphi \mid \sigma \rightsquigarrow_i \psi$$

The semantics and the truth definitions of the formulas are defined as earlier with the exception of strategy specifications for restrictions (cf. Section 2). The axiom system of RSL consists of the axioms and rules of SL together with the following additional axiom for the added specification construct.

$$- (\sigma!a)_i : c \leftrightarrow \mathbf{turn}_i \wedge \neg((\sigma)_i : a) \wedge (\sigma)_i : c$$

The soundness of the given axiom is straightforward and hence skipped. The new specification we introduced does not bring along an extra derivation rule since the new operator is at the level of specifications, not the formulas.

Now, we observe that restricted moves are not enabled. We state it as a theorem with an immediate proof.

**Theorem 2.**  $(\sigma!a)_i : a \leftrightarrow \perp$ .

As expected, the system RSL is complete.

**Theorem 3.** *RSL is complete with respect to the given semantics.*

However, the decidability of RSL and SL are not immediate. To the best of our knowledge, SL has not yet been proved to be decidable (or undecidable). Here, we show an upper bound for the model checking problem for both logics.

Now, we discuss a reduction of SL to (multi-modal) Computational Tree Logic CTL\*. We refer the readers who are not familiar with CTL\* to [9,10]. First, we discuss how to construct a CTL\* model based on a given SL/RSL model, then describe a translation from SL to CTL\*.

Let us take a strategy model  $M = (T, V)$  where  $T = (S, \Rightarrow, s_0, \lambda)$ . Notice that the function  $\Rightarrow$  was defined from  $S \times \Sigma$  to  $S$ . We can *redefine* it by *currying*. Given  $\Rightarrow: S \times \Sigma \rightarrow S$ , we can define the transition  $\overset{a}{\Rightarrow}: S \rightarrow S$  for each move  $a$ .



Therefore, we can curry  $\Rightarrow$  to get  $\Rightarrow = \bigcup_{a \in \Sigma} \overset{a}{\Rightarrow}$ . We can think of  $M$  as a pointed multi-modal CTL\* tree model  $M^* = (S, s_0, \{\overset{a}{\Rightarrow}\}_{a \in \Sigma}, V)$  that has next time modalities for each action. In this case, corresponding to each binary relation  $\overset{a}{\Rightarrow}$ , we will have a dynamic next-time modality  $X_a$  which quantifies over the sub-path on the same branch (note that state formulas are also path formulas [10]). For the reflexive-transitive closure of  $\overset{a}{\Rightarrow}$ , we will use  $\Box$  for all accessible future times in the same branch. Before giving the translation of SL specifications and formulas into CTL\*, let us introduce some special propositions. We label the states that are returned by a strategy  $\mu$  with the proposition **strategy** $_\mu$ , stipulating that **strategy** $_\mu$  holds only at those points. Notice that given two points in the domain of  $\mu$ , there is a unique path between these two (which satisfies **strategy** $_\mu$ ). Moreover, we use **turn** $_i$  as a proposition that denotes that it is  $i$ 's turn to play, i.e.  $s \models \text{turn}_i$  iff  $s \in S^i$ .

We now give two translations. First, **tr** translates strategy specifications to CTL\* formulas while the second **Tr** translates SL formulas to CTL\* formulas. Given a strategy  $\mu$ , conformation to  $\mu$  is translated as follows.

- $\text{tr}([\psi \rightarrow a]^i) = \text{Tr}(\psi) \rightarrow E(\text{strategy}_\mu \wedge X_a \top)$
- $\text{tr}(\sigma_1 + \sigma_2) = \text{tr}(\sigma_1) \vee \text{tr}(\sigma_2)$
- $\text{tr}(\sigma_1 \cdot \sigma_2) = \text{tr}(\sigma_1) \wedge \text{tr}(\sigma_2)$

Notice that in the first case, the translation makes use of the formula translation **Tr** for formula  $\psi$  as defined below. Assuming the correctness of **Tr** (which we show in Theorem 5), correctness of the translation **tr** is straightforward.

**Theorem 4.** *Let  $\mu$  be a strategy and  $\sigma$  a strategy specification in SL, and let  $\mu^*$  be the corresponding (sub)tree in a CTL\* model. Then,  $\mu, s \models \sigma$  iff  $\mu^*, s \models \text{tr}(\sigma)$ .*

Here follows the translation **Tr** of formulas from SL to CTL\* skipping the Boolean cases. Note that the translation is very similar to the Kripke semantics for Propositional Dynamic Logic where for each action  $a$ , a relation  $R_a$  and a modality  $\langle a \rangle$  associated with  $R_a$  are introduced.

- $\text{Tr}(\langle a \rangle \varphi) = X_a \text{Tr}(\varphi)$
- $\text{Tr}((\sigma)_i : c) = X_c \top$  for  $c \in \sigma(s)$
- $\text{Tr}((\sigma)_i : c) = \perp$  if  $c \notin \sigma(s)$
- $\text{Tr}(\sigma \rightsquigarrow_i \psi) = E\Box(\text{strategy}_\sigma \wedge (\text{Tr}(\psi) \wedge (\text{turn}_i \rightarrow \text{enabled}_\sigma)))$

Now, the atom **enabled** $_\sigma$  is true at a state  $s$  in CTL\* if and only if for at least one  $a \in \Sigma$  we have  $X_a \top$  and  $a \in \sigma(s)$ . Finally, the CTL\* correspondence of  $\sigma(s)$  to the set of enabled moves at  $s$  by the strategy specification  $\sigma$  is defined exactly as before with one small arrangement in the definition of admissible moves at a given state  $s$ , namely  $\text{moves}(s) = \{a : s \models_{CTL^*} X_a \top\}$ . As an example, consider the translation of the proposition **out** $_\mu = a$ . Recall that it means that  $a$  is the unique outgoing edge according to strategy  $\mu$  at the state where the formula is interpreted. Therefore, there is a branch that is followed by strategy  $\mu$ , and at that branch, at the current state,  $a$  is an admissible move. This corresponds to the translation  $E(\text{strategy}_\mu \wedge X_a \top)$ . The following theorem summarizes our efforts here.

**Theorem 5.** *Let  $M$  be a SL model and let  $M^*$  be its CTL\* correspondent. Then,  $M, s \models \varphi$  iff  $M^*, s \models \mathbf{Tr}(\varphi)$  for any state  $s \in S$ .*

Note that the translation we suggest is model-dependent. For example, the predicate **strategy** <sub>$\mu$</sub> , depends on the strategy  $\mu$ , thus the model. For this reason, the suggested translation is not entirely syntactic and does not give us an immediate decidability result (using the fact that CTL\* is decidable). However, model checking problem for CTL\* is PSPACE-complete [9]. Therefore, we have an upper bound for the complexity of the model checking problem for SL. We next observe that the model checking problem for both SL and RSL are in PSPACE.

**Theorem 6.** *The model checking problem for SL is in PSPACE.*

**Corollary 1.** *The model checking problem for RSL is in PSPACE.*

## 4 Related Work and Conclusion

From a logical point of view, the idea of treating strategies as the “unsung heroes of games” can be traced back to van Benthem [4,5]. In line with this program, Ramanujam, Simon, and Paul [15,18,19,20] have initiated a study within game theory and dynamic logic by taking strategies as the focus of concern and treating them as the primitives of games. They discuss strategies as a way of reasoning within and about games, and investigate how games and strategies behave under some further assumptions.

The game-theoretical approach to the centipede game was initiated by Rosenthal [21], which was followed by several solution methods [2,13]. More recently, the role of rationality in its solution methods has been discussed [3,6] and extended to some other cases [16]. The role knowledge as opposed to beliefs has been discussed in similar contexts by Artemov [1]. Moreover, based on experimental work, Ghosh, Meijering and Verbrugge aim to observe how humans reason strategically [11]. They consider simple centipede-like games from the cognitive point of view, and use strategy logic to formalize their findings.

This work is built on the aforementioned studies, and introduces a dynamic twist in formalizing strategies in games. What we suggest is a formal framework where restrictions in strategies are allowed during the game. We propose a new logic, called restricted strategy logic (RSL), to express strategy restrictions. We show its completeness and discuss model checking issues. Along the way, we also discuss model checking of SL, which has not been done so far (to the best of our knowledge). We claim that RSL presents a succinct way to represent strategy revisions. One direction for future research concerns the question on how RSL can be connected to the switching strategy frameworks. Additionally, one could come up with a restriction methodology where players change their rationality assumption. Note that, within the framework of RSL, different types of rationality (utilitarian, deontological etc) can be seen as a different set of *restrictions*. For instance, a player can initially commit herself to a maxmin type of strategy and then change her commitment to a maxmax type of strategy.

Therefore, such changes can be represented from switching from restriction set  $A$  to restriction set  $A'$ , for instance. In short, different *types* of rationality can cause different revisions in players' strategies. We leave such analysis to future work.

Moreover, note that the way strategy restrictions work, and its completeness proof resemble public announcement logic [8,17]. In the case of RSL, the move restriction can be seen as a prohibitive *negative* announcement, corresponding to an elimination of moves that agree with the prohibition.

In conclusion, we believe that RSL presents a concise and natural framework for dynamic strategizing in games, and it can be extended in several thought-provoking ways.

*Acknowledgements* We especially acknowledge the help and encouragement of Sujata Ghosh and Rineke Verbrugge. Sujata spent hours reading the paper and providing me with feedback, and Rineke read the paper very carefully many times and suggested lots of corrections. The idea of using CTL\* for the decidability result and the name “restricted strategy logic” have been suggested by Sujata. This paper is the product of author's visit to the Department of Artificial Intelligence of the University of Groningen. We also thank Sergei Artemov and Rohit Parikh for their feedback.

## References

1. Sergei Artemov. Rational decisions in non-probabilistic setting. Technical Report TR-2009012, Department of Computer Science, The Graduate Center, The City University of New York, 2009.
2. R. J. Aumann. Backward induction and common knowledge of rationality. *Games and Economic Behavior*, 8(1):6–19, 1995.
3. A. Baltag, S. Smets, and J. Zvesper. Keep ‘hoping’ for rationality: A solution to the backward induction paradox. *Synthese*, 169(2):301–333, 2009.
4. J. van Benthem. Extensive games as process models. *Journal of Logic, Language and Information*, 11:289–313, 2002.
5. J. van Benthem. In praise of strategies. In J. van Eijck and R. Verbrugge, editors, *Games, Actions and Social Software*, Texts in Logics and Games. Springer Verlag, Berlin, 2011.
6. J. van Benthem and A. Gheerbrant. Game solution, epistemic dynamics and fixed-point logics. *Fundamenta Informaticae*, 100:19–41, January 2010.
7. Patrick Blackburn, Maartijn de Rijke, and Yde Venema. *Modal Logic*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2001.
8. H. van Ditmarsch, W. van der Hoek, and B. Kooi. *Dynamic Epistemic Logic*. Springer, Berlin, 2007.
9. A.E. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 995–1072. Elsevier and MIT-Press, Amsterdam and Cambridge (MA), 1990.
10. E.A. Emerson and J.Y. Halpern. Sometimes’ and ‘not never’ revisited: On branching versus linear time temporal logic. *Journal of the Association for Computing Machinery*, 33:151–178, 1986.

11. S. Ghosh, B. Meijering, and R. Verbrugge. Logic meets cognition: Empirical reasoning in games. In O. Boissier et al., editor, *MALLOW*, volume 627 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2010.
12. S. Ghosh, R. Ramanujam, and S. Simon. On strategy composition and game composition. Unpublished manuscript, 2010.
13. J. Y. Halpern. Substantive rationality and backward induction. *Games and Economic Behavior*, 37(2):425–435, November 2001.
14. Wilfred Hodges. Logic and games. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. 2009.
15. S. Paul, R. Ramanujam, and S. Simon. Dynamic restriction of choices: a preliminary logical report. In A. Heifetz, editor, *TARK*, pages 218–226, 2009.
16. M. Piccione and A. Rubinstein. On the interpretation of decision problems with imperfect recall. In Y. Shoham, editor, *TARK*, pages 75–76. Morgan Kaufmann, 1996.
17. J. A. Plaza. Logics of public communication. In M. L. Emrich, M. S. Pfeifer, M. Hadzikadic, and Z. W. Ras, editors, *Proceedings 4th International Symposium on Methodologies for Intelligent Systems*, pages 201–216. Oak Ridge National Laboratory, ORNL/DSRD-24, 1989.
18. R. Ramanujam and S. Simon. Dynamic logic on games with structured strategies. In *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR-08)*, pages 49–58. AAAI Press, 2008.
19. R. Ramanujam and S. Simon. A logical structure for strategies. In *Logic and the Foundations of Game and Decision Theory (LOFT 7)*, volume 3 of *Texts in Logic and Games*, pages 183–208. Amsterdam University Press, 2008.
20. R. Ramanujam and S. Simon. Reasoning in games. In M. K. Chakraborty, B. Löwe, and M. N. Mitra, editors, *Logic, Navya-Nyaya and its Applications: Homage to Bimal Krishna Chakraborty*. College Publications, London, 2008.
21. R. W. Rosenthal. Games of perfect information, predatory pricing and the chain-store paradox. *Journal of Economic Theory*, 25(1):92–100, 1981.
22. Ulrich Schwalbe and Paul Walker. Zermelo and the early history of game theory. *Games and Economic Behavior*, 34(1):123–137, January 2001.

## Appendix: Proofs

*Proof (Proposition 2).* Consider the first case.

$$\begin{aligned}
\mu, s \models (\sigma \cdot \sigma')!a \quad &\text{iff} \quad a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma \cdot \sigma' \\
&\text{iff} \quad a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma \text{ and } \mu!a, s \models \sigma' \\
&\text{iff} \quad a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma), \text{ and} \\
&\quad a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma') \\
&\text{iff} \quad \mu, s \models \sigma!a \text{ and } \mu, s \models \sigma'!a
\end{aligned}$$

The remaining cases are similar. □

*Proof (Proposition 3).* We start with considering the cases where the restrictions are applied consecutively twice. We use Proposition 2.

$$\begin{aligned}
\mu, s \models (\sigma!a)!a \quad &\text{iff} \quad a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma!a \\
&\text{iff} \quad a \notin \mathbf{outr}_\mu(s) \text{ and } (\mathbf{out}_\mu(s) \neq a \text{ and } (\mu!a)!a, s \models \sigma) \\
&\text{iff} \quad a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma \\
&\text{iff} \quad \mu, s \models \sigma!a
\end{aligned}$$

Now, we generalize to the case for  $n$ . The proof is by induction on  $n$ . The case for  $n = 1$  is trivial and the case for  $n = 2$  was presented above. Assume now that  $n \geq 2$  and the claim holds for every integer less than or equal to  $n$ . We will now show it for  $n + 1$ .

$$\begin{array}{lll}
\mu, s \models \sigma!^n a & \text{iff} & \\
\mu, s \models (\sigma!^{n-1} a)!a & \text{iff} & a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma!^{n-1} a \\
& & \text{iff } a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma!^a \quad \text{induction hyp.} \\
& & \text{iff } \mu, s \models \sigma!a \quad \text{by definition}
\end{array}$$

□

*Proof (Proposition 4).* First, notice that for a strategy  $\mu$  and moves  $a, b$ , we have  $(\mu!a)!b = (\mu!b)!a$  by definition. Now, let us consider strategy specification  $\sigma$  with moves  $a$  and  $b$ .

$$\begin{array}{ll}
\mu, s \models (\sigma!a)!b & \text{iff } b \notin \mathbf{outr}_\mu(s) \text{ and } \mu!b, s \models \sigma!a \\
& \text{iff } b \notin \mathbf{outr}_\mu(s) \text{ and } (a \notin \mathbf{outr}_\mu(s) \text{ and } (\mu!b)!a, s \models \sigma) \\
& \text{iff } b \notin \mathbf{outr}_\mu(s) \text{ and } (a \notin \mathbf{outr}_\mu(s) \text{ and } (\mu!a)!b, s \models \sigma) \\
& \text{iff } a \notin \mathbf{outr}_\mu(s) \text{ and } (b \notin \mathbf{outr}_\mu(s) \text{ and } (\mu!a)!b, s \models \sigma) \\
& \text{iff } a \notin \mathbf{outr}_\mu(s) \text{ and } \mu!a, s \models \sigma!b \\
& \text{iff } \mu, s \models (\sigma!b)!a
\end{array}$$

□

*Proof (Theorem 3).* The completeness of RSL is by reduction to SL. Since we have one additional strategy specification, we describe an immediate reduction for that.

First notice that an RSL model is a submodel of a suitable SL model. The only problem is that, in RSL, strategies are constructed as relations whereas in SL, the strategies are functions. Therefore, an RSL strategy can be thought of as a union of several SL strategies. Therefore, a RSL game-tree is a tree model for a SL model with the suitable union of strategies. Moreover, in RSL, we also obtain a game-tree model with relational strategies.

Now, the reduction of RSL to SL should be rather immediate. Given a formula of the form  $(\sigma)_i : c$  where  $\sigma = \sigma'!a$  for some  $a$ , we observe that (by soundness), it is equivalent to a formula in the language of SL:  $\mathbf{turn}_i \wedge \neg((\sigma)_i : a) \wedge (\sigma)_i : c$ . Similarly, consider the given formula of the form  $\sigma \rightsquigarrow_i \psi$  where  $\sigma = \sigma'!a$  for some  $a$ . Notice also that, in SL,  $\sigma \rightsquigarrow_i \psi$  is axiomatically reduced to a formula that uses formulas of the form  $(\sigma)_i : a$  which we have covered just before. Therefore, all two different types of formulas that may include restricted strategy specifications are reduced to a formula in the language of SL. Since SL is complete already [12], and our translation is truth preserving due to soundness, we conclude that RSL is complete. □

*Proof (Theorem 4).* Let  $\mu : S \rightarrow \Sigma$  be a strategy and let  $\sigma$  be a strategy specification in SL for a fixed player  $i$ . For simplicity, we omit the superscripts that indicate the player. First, we describe how to obtain a multi-modal CTL\* tree  $\mu^*$ , and then show that the translation is truth-preserving. As a reminder, in a strategy tree, we include the root, and for the states that belong to the strategizing

player, we assign a unique move to the player. In addition, we include all other moves that do not belong to the strategizing player. Therefore, a strategy tree can be thought of as a branching multi-modal CTL\* model. We put  $w \models \mathbf{turn}_i$  if  $w \in S^i$  and at each  $w \in S^i$ , we have one outgoing edge. For  $v \notin S^i$ , we have all admissible moves  $\mu(v)$  at  $v$ . Then, the CTL\* model is constructed as follows. Take  $S$  as the set of states of the CTL\* model  $\mu^*$ . Next, for all moves  $a \in \Sigma$ , we have a corresponding accessibility relation  $R_a$  in the CTL\* model similar to the Kripke semantics for propositional dynamic logic. Finally, we keep the valuation the same as in the SL model. Now, let us consider the case  $\sigma = [\psi \rightarrow a]^i$ :

$$\begin{aligned} \mu, s \models \sigma & \text{ iff } \mu, s \models [\psi \rightarrow a]^i \\ & \text{ iff } \mu, s \models \psi \text{ implies } \mathbf{out}_\mu = a \quad (\text{by definition}) \\ & \text{ iff } \mu^*, s \models \mathbf{Tr}(\psi) \text{ implies } \mathbf{E}(\mathbf{strategy}_\mu \wedge \mathbf{X}_a \top) \end{aligned}$$

Here we make use of the formula translation  $\mathbf{Tr}$  whose correctness will be shown next. Furthermore, we obtain the last line by the induction hypothesis, and by the earlier observation we have made about  $\mathbf{out}_\mu = a$ . The remaining cases for the translation  $\mathbf{tr}$  are straightforward inductions on specifications  $\sigma_1 + \sigma_2$  and  $\sigma_1 \cdot \sigma_2$ , and hence left to the reader. Unconventionally, here we make use of Theorem 5 in the proof of the above statement just because specifications precede the formulas in SL.  $\square$

*Proof (Theorem 5).* Take a SL model  $M$  and the corresponding multi-modal CTL\* model  $M^*$ . We then have the following:

$$\begin{aligned} M, s \models \langle a \rangle \varphi & \text{ iff } \exists s' \text{ s.t. } s \xrightarrow{a} s' \text{ and } M, s' \models \varphi \quad (\text{by definition}) \\ & \exists s' \text{ s.t. } s \xrightarrow{a} s' \text{ and } M^*, s' \models \mathbf{Tr}(\varphi) \quad (\text{by induction}) \\ M^*, s \models \mathbf{X}_a \mathbf{Tr}(\varphi) & \quad (\text{by definition}) \end{aligned}$$

Similarly, consider the SL formula  $\sigma \rightsquigarrow_i \psi$ :

$$\begin{aligned} M, s \models \sigma \rightsquigarrow_i \psi & \text{ iff for all } s' \text{ such that } s \Rightarrow_\sigma^* s' \text{ in } T_s|_\sigma, \text{ we have,} \\ & M, s' \models \psi \wedge (\mathbf{turn}_i \rightarrow \mathbf{enabled}_\sigma) \\ & \quad (\text{by definition}) \\ & \text{ iff for all } s' \text{ such that } s \Rightarrow^* s' \text{ in } T_s, \text{ we have} \\ & M, s' \models \mathbf{strategy}_\sigma \wedge \psi \wedge (\mathbf{turn}_i \rightarrow \mathbf{enabled}_\sigma) \\ & \quad (\text{by definition}) \\ & \text{ iff for all } s' \text{ such that } s \Rightarrow^* s' \text{ in } T_s, \text{ we have} \\ & M^*, s' \models \mathbf{strategy}_\sigma \wedge \mathbf{Tr}(\psi) \wedge (\mathbf{turn}_i \rightarrow \mathbf{enabled}_\sigma) \\ & \quad (\text{by induction}) \\ & \text{ iff } M^*, s \models \mathbf{E}\Box[\mathbf{strategy}_\sigma \wedge \mathbf{Tr}(\psi) \wedge (\mathbf{turn}_i \rightarrow \mathbf{enabled}_\sigma)] \\ & \quad (\text{as } \Rightarrow^* \text{ denotes the reflexive and transitive closure} \\ & \quad \text{on the path } T_s \text{ on the CTL* tree } T) \end{aligned}$$

The last case  $(\sigma)_i : c$  is very similar. Recall that this formula is true in SL at a state  $s$  iff  $c \in \sigma(s)$ . Consider the enabled move  $c$  at  $s$ , and take a corresponding next time modality for  $c$ , namely  $\mathbf{X}_c$ . Moreover, at  $s$ , the diamond-like modality  $\mathbf{X}_c$  has to be *enabled* giving us  $\mathbf{X}_c \top$  for  $c \in \sigma(s)$ .  $\square$

*Proof (Theorem 6).* In this proof, we will present two different arguments to show that the model checking problem for SL is in PSPACE.

First, we show that the translation  $\mathbf{Tr}$ : SL  $\rightarrow$  multi-modal CTL\* is polynomial-time in terms of the length of the formulas. We will denote the length of a specification or a formula by  $|\cdot|$ . We will show that  $|\mathbf{Tr}(\psi)| \leq |\psi|^k$  for some integer  $k$ . The cases for Booleans are obvious, hence we skip them. For  $\psi = \langle a \rangle \varphi$ , consider  $|\mathbf{Tr}(\langle a \rangle \varphi)|$  which is equivalent to  $|\mathbf{X}_a \mathbf{Tr}(\varphi)| = 1 + |\mathbf{Tr}(\varphi)|$ . By induction hypothesis,  $|\mathbf{Tr}(\varphi)| \leq |\varphi|^l$  for some integer  $l$ . Therefore,  $1 + |\mathbf{Tr}(\varphi)| \leq |\varphi|^k$  for some  $k \geq 1 + l$ . Therefore, for  $\psi = \langle a \rangle \varphi$ , we observe  $|\mathbf{Tr}(\psi)| \leq |\psi|^k$  for some integer  $k$ . In a similar fashion, the case for  $\psi = (\sigma)_i : c$  is obvious as  $|\mathbf{Tr}((\sigma)_i : c)|$  is always constant. The case for  $\psi = \sigma \rightsquigarrow_i \varphi$  is also very similar. By induction hypothesis, we immediately observe that  $|\mathbf{Tr}(\sigma \rightsquigarrow_i \varphi)| \leq 11 + |\sigma \rightsquigarrow_i \varphi|^l$  (counting the parantheses as well). Therefore, for some large enough integer  $k > l$ , we have  $|\mathbf{Tr}(\sigma \rightsquigarrow_i \varphi)| \leq |\sigma \rightsquigarrow_i \varphi|^k$ .

Now, we can consider the translation  $\mathbf{tr}$  for the strategy specifications. The reason why we consider  $\mathbf{tr}$  after  $\mathbf{Tr}$  is the fact that former depends on the latter. We will now show that, for strategy specification  $\sigma$ ,  $|\mathbf{tr}(\sigma)| \leq |\sigma|^k$  for some integer  $k$ . Consider the case where  $\sigma = [\psi \rightarrow a]^i$ . Then,  $|\mathbf{tr}([\psi \rightarrow a]^i)| = 5 + |\mathbf{Tr}(\psi)|$ . By the previous observation, we know that  $|\mathbf{Tr}(\psi)| \leq |\psi|^l$  for some integer  $l$ . Therefore,  $5 + |\mathbf{Tr}(\psi)| \leq |\psi|^k$  for large enough integer  $k \geq l$ . Thus,  $\mathbf{tr}(\psi \rightarrow a)^i \leq |(\psi \rightarrow a)^i|^k$  for some  $k$ . The cases for the  $\cdot$  and  $+$  operations are obvious. This concludes the proof that the translation functions  $\mathbf{tr}$  and  $\mathbf{Tr}$  are polynomial-time. Therefore, the complexity of model checking for SL cannot be higher than PSPACE.

Our second argument is a sketch of a direct reasoning. Similar to the arguments for the complexity of basic modal logic, we just need to check the branches of the tree model one by one using a depth-first search [7]. Now, fix a SL formula  $\varphi$  and a model  $M$ . Since SL models are branching tree models, we can check  $M$ , branch by branch, one at a time without any need of considering the other branches. Namely, the procedure does not need to *remember* the previous searches making it effective in the use of space. Moreover, the length of the branches in  $M$  is polynomial in  $|\varphi|$  because of the construction of the SL model  $M$ , thus it shows that model checking for SL is in PSPACE. This concludes the proof that the complexity of the model checking problem for SL is in PSPACE.  $\square$

*Proof (Corollary 1).* We observed that the RSL formulas can be reduced to SL formulas. The reduction is clearly polynomial, as can be seen from the axiomatization (See Section 3.3). Therefore, we can translate any given RSL formula to a SL in a SL model, which in turn can be translated into a multi-modal CTL\* formula and model. Then, by Theorem 6, we deduce that the complexity of the model checking problem for RSL is also PSPACE.  $\square$